

Particle Gibbs algorithms for Markov jump processes

Błażej Miasojedow

Wojciech Niemiro

Institute of Applied Mathematics, University of Warsaw

Banacha 2, 02-097 Warsaw, Poland

BMIA@MIMUW.EDU.PL

WNIEM@MIMUW.EDU.PL

Editor:

Abstract

In the present paper we propose a new MCMC algorithm for sampling from the posterior distribution of hidden trajectory of a Markov jump process. Our algorithm is based on the idea of exploiting virtual jumps, introduced by [Rao and Teh \(2013\)](#). The main novelty is that our algorithm uses particle Gibbs with ancestor sampling (PGAS, see [Andrieu et al. \(2010\)](#); [Lindsten et al. \(2014b\)](#)) to update the skeleton, while Rao and Teh use forward filtering backward sampling (FFBS). In contrast to previous methods our algorithm can be implemented even if the state space is infinite. In addition, the cost of a single step of the proposed algorithm does not depend on the size of the state space. The computational cost of our method is of order $\mathcal{O}(NE(n))$, where N is the number of particles used in the PGAS algorithm and $\mathbb{E}(n)$ is the expected number of jumps (together with virtual ones). The cost of the algorithm of Rao and Teh is of order $\mathcal{O}(|\mathcal{X}|^2\mathbb{E}(n))$, where $|\mathcal{X}|$ is the size of the state space. Simulation results show that our algorithm with PGAS converges slightly slower than the algorithm with FFBS, if the size of the state space is not big. However, if the size of the state space increases, the proposed method outperforms existing ones. We give special attention to a hierarchical version of our algorithm which can be applied to continuous time Bayesian networks (CTBNs).

Keywords: Continuous time Markov processes, Bayesian networks, MCMC, Sequential Monte Carlo, Hidden Markov models, Posterior sampling, CTBN

1. Introduction

Markov jump processes (MJP) are natural extension of Markov chains to continuous time. They are widely applied in modelling of the phenomena of chemical, biological, economic and other sciences. An important class of MJP are continuous time Bayesian networks (CTBN) introduced by [Schweder \(1970\)](#) under the name of composable Markov chains and then reinvented by [Nodelman et al. \(2002a\)](#) under the current name. Roughly, a CTBN is a multivariate MJP in which the dependence structure between coordinates can be described by a graph. Such a graphical representation allows for decomposing a large intensity matrix into smaller conditional intensity matrices.

In many applications it is necessary to consider a situation where the trajectory of a Markov jump process is not observed directly, only partial and noisy observations are available. Typically, the posterior distribution over trajectories is then analytically intractable.

The present paper is devoted to MCMC methods for sampling from the posterior in such a situation.

In the literature there exist several approaches to the above mentioned problem: based on sampling (Boys et al., 2008; El-Hay et al., 2008; Fan and Shelton, 2008; Fearnhead and Sherlock, 2006; Hobolth and Stone, 2009; Nodelman et al., 2002b; Rao and Teh, 2013, 2012; Miasojedow et al., 2014), based on numerical approximations (Cohn et al., 2010; Nodelman et al., 2002a, 2005; Oppen and Sanguinetti, 2008). Some of these methods are inefficient, like modification of likelihood weighting (Nodelman et al., 2002b). Other approaches involve expensive computations like matrix exponentiation, spectral decomposition of matrices, finding roots of equations. There are also approximate algorithms based on time discretization. To the best of our knowledge the most general, efficient and exact method is that proposed by Rao and Teh (2013), and extended to a more general class of continuous time discrete systems in Rao and Teh (2012). Their algorithm is based on introducing so-called virtual jumps and a thinning procedure for Poisson processes. In our approach we combine this method with particle MCMC discovered by Andrieu et al. (2010). More precisely, instead of forward filtering backward sampling algorithm used in the original version, we use particle Gibbs (Andrieu et al., 2010) with added ancestor resampling proposed in Lindsten et al. (2012, 2014b). The proposed method is computationally less expensive. Moreover, our algorithm can be directly applied when the state space is infinite, in opposition to Rao and Teh (2012, 2013).

2. Markov jump processes

Consider a continuous time stochastic process $\{X(t), t \geq 0\}$ defined on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ with a discrete state space \mathcal{X} . Assume the process is time-homogeneous Markov with transition probabilities

$$P^t(s, s') = \mathbb{P}(X(t+u) = s' | X(u) = s) ,$$

for $s, s' \in \mathcal{X}$. The initial distribution is denoted by $\nu(s) = \mathbb{P}(X(0) = s)$. Since \mathcal{X} is discrete, ν can be viewed as a vector and P^t as a matrix (both possibly infinite). The intensity matrix is defined as follows

$$Q(s, s') = \lim_{t \rightarrow 0} \frac{1}{t} [P^t(s, s') - I(s, s')] ,$$

where $I = P^0$ is the identity matrix. Equivalently, $Q(s, s')$ is the intensity of jumps from s to s' , i.e.

$$\begin{aligned} \mathbb{P}(X(t+dt) = s' | X(t) = s) &= Q(s, s')dt \text{ for } s \neq s' , \\ \mathbb{P}(X(t+dt) = s | X(t) = s) &= 1 - Q(s)dt , \end{aligned}$$

where $Q(s) = -Q(s, s) = \sum_{s' \neq s} Q(s, s')$ denotes the intensity of leaving state s . Clearly we have $\sum_{s'} Q(s, s') = 0$ for all $s \in \mathcal{X}$. Throughout this paper we assume that Q is non-explosive (Norris, 1998), which means that almost surely only finite number of jumps occur in any bounded time interval $[0, t_{\max}]$. This assumption is fulfilled in most applications we have in mind. Sufficient and necessary conditions for Q to be non-explosive can be found in (Norris, 1998)[Thm. 2.7.1, 2.7.2, Cor. 2.7.3]. From now on the interval $[0, t_{\max}]$ is fixed. Let

there be m jumps and let these jumps occur at ordered moments $T = (t_1, \dots, t_m)$. Moments of jumps T with a corresponding sequence of states, denoted by $S = (s_0, s_1, \dots, s_m) = (X(0), X(t_1), \dots, X(t_m))$, fully describe the trajectory $X([0, t_{\max}])$. By definition of the process $\{X(t)\}$, every interval between jumps, $t_j - t_{j-1}$ for $j = 1, \dots, m$, with the convention $t_0 = 0$, is distributed according to the exponential distribution with parameter $Q(s_{j-1})$. The skeleton S is a discrete time Markov chain with initial distribution ν and transition matrix given by

$$\begin{cases} \frac{Q(s, s')}{Q(s)} & \text{if } s \neq s' ; \\ 0 & \text{if } s = s' . \end{cases}$$

Thus random variable (T, S) has density

$$\begin{aligned} p(T, S) &= \nu(s_0) \prod_{j=1}^m Q(s_{j-1}) \exp \{-Q(s_j)(t_j - t_{j-1})\} \frac{Q(s_{j-1}, s_j)}{Q(s_{j-1})} \exp \{-Q(s_m)(t_{\max} - t_m)\} \\ &= \nu(s_0) \prod_{j=1}^m Q(s_{j-1}, s_j) \exp \{-Q(s_j)(t_j - t_{j-1})\} \exp \{-Q(s_m)(t_{\max} - t_m)\} , \end{aligned} \quad (1)$$

where $m = |T|$. The last factor $\exp \{-Q(s_m)(t_{\max} - t_m)\}$ comes from the fact that the waiting time for jump $m + 1$ can be arbitrary but greater than $t_{\max} - t_m$.

3. Virtual jumps

Let $\{X(t)\}$ be a homogeneous Markov process with intensity matrix Q and let $R(s) \geq Q(s)$ for all $s \in \mathcal{X}$. Consider the following sampling scheme (Rao and Teh, 2012), based on dependent thinning, i.e. rejection sampling for an inhomogeneous Poisson process (Lewis and Shedler, 1979). We generate a sequence of potential times of jumps $(\tilde{t}_1, \tilde{t}_2, \dots)$. For a given moment \tilde{t}_{k-1} and a current value of the process $\tilde{s}_{k-1} = X(\tilde{t}_{k-1})$, we draw the next time interval $\tilde{t}_k - \tilde{t}_{k-1}$ from the exponential distribution with parameter $R(\tilde{s}_{k-1})$. With probability $Q(\tilde{s}_{k-1})/R(\tilde{s}_{k-1})$ the process jumps at time \tilde{t}_k to another state, and this new state is \tilde{s}_k with probability $Q(\tilde{s}_{k-1}, \tilde{s}_k)/Q(\tilde{s}_{k-1})$. With probability $(1 - Q(\tilde{s}_{k-1})/R(\tilde{s}_{k-1}))$ the process does not jump and we put $\tilde{s}_k = \tilde{s}_{k-1}$. The resulting redundant skeleton $\tilde{S} = (\tilde{s}_0, \tilde{s}_1, \tilde{s}_2, \dots)$ is therefore a Markov chain with transition matrix P defined by

$$P(s, s') = \begin{cases} \frac{Q(s, s')}{R(s)} & \text{if } s \neq s' ; \\ 1 - \frac{Q(s)}{R(s)} & \text{if } s = s' . \end{cases} \quad (2)$$

We summarize this procedure as the following algorithm 1.

Algorithm 1 Thinning procedure.

Set $\tilde{t}_0 = 0$ and $k = 0$.
 Draw $\tilde{s}_0 \sim \nu(\cdot)$.
while $\tilde{t}_k < t_{\max}$ **do**
 Set $k = k + 1$.
 Draw $W \sim \text{Exp}(R(\tilde{s}_{k-1}))$.
 Set $\tilde{t}_k = \tilde{t}_{k-1} + W$.
 Draw $\tilde{s}_k \sim P(\tilde{s}_{k-1}, \cdot)$.
end while

As before, consider the process in a fixed interval of time $[0, t_{\max}]$. Let $\tilde{T} = (\tilde{t}_1, \tilde{t}_2, \dots, \tilde{t}_n)$ be the set of moments generated by the above algorithm. Let $J = \{k > 1 : \tilde{s}_k \neq \tilde{s}_{k-1}\}$. Denote by $T = \tilde{T}_J$ and $V = \tilde{T}_{-J}$ the moments of true jumps and virtual jumps, respectively. The process $\{X(t)\}$ resulting from the algorithm has the same probability distribution as that in Section 2. This fact is explicitly formulated in Proposition 1 below.

Proposition 1 *The marginal distribution of $(T = \tilde{T}_J, \tilde{S}_J)$ has the density given by (1).*

This proposition is known and can be found e.g. in Rao and Teh (2012). However, to make the paper self-contained we give a proof in the Appendix. The following corollary shows that, conditionally on (T, \tilde{S}_J) , i.e. on true jumps and the skeleton, the set of virtual jumps V is an inhomogeneous Poisson process with intensity $R(X(t)) - Q((X(t)))$.

Corollary 2 *Let V_j denote moments of virtual jumps between two adjacent true jumps t_{j-1} and t_j . The conditional density of V_j is given by*

$$p(V_j | X(t_{j-1}) = s, t_{j-1}, t_j) = (R(s) - Q(s))^{|V_j|} \exp \{-(t_j - t_{j-1})(R(s) - Q(s))\}.$$

The proof is also given in the Appendix. In the sequel we will work with the redundant representation of the process $\{X(t)\}$ introduced in this section. For simplicity, let us slightly abuse notation and from now on write S instead of \tilde{S} . Clearly, the density of (T, V, S) is given by

$$\begin{aligned} p(T, V, S) &= \nu(s_0) \prod_{k=1}^n R(s_{k-1}) P(s_{k-1}, s_k) \exp \left\{ - \int_0^{t_{\max}} R(X(u)) du \right\} \\ &= \nu(s_0) \prod_{k=1}^n (R(s_{k-1}) - Q(s_{k-1}))^{\mathbf{1}(s_k = s_{k-1})} Q(s_{k-1}, s_k)^{\mathbf{1}(s_k \neq s_{k-1})} \\ &\quad \times \prod_{k=1}^n \exp \{ -R(s_k)(\tilde{t}_k - \tilde{t}_{k-1}) \} \exp \{ -R(s_n)(t_{\max} - \tilde{t}_n) \}, \end{aligned} \tag{3}$$

where $n = |T| + |V|$ is the total number of jumps.

Now we describe two particular choices of intensity R . The first one leads to the so-called uniformization (Jensen, 1953; Cinlar, 1975; Hobolth and Stone, 2009; Rao and Teh, 2013). Let $\lambda \geq \max_s Q(s)$ and consider moments of potential jumps distributed according to homogeneous Poisson process with intensity λ . Precisely, we consider the thinning procedure

with $R(s) \equiv \lambda$. Clearly, the joint distribution of true jumps, virtual jumps and skeleton is now given by

$$p(T, V, S) \propto \lambda^n \nu(s_0) \prod_{k=1}^n P(s_{k-1}, s_k), \quad (4)$$

where P is the transition matrix of discrete time Markov chain defined by

$$P(s, s') = \begin{cases} \frac{Q(s, s')}{\lambda} & \text{if } s \neq s'; \\ 1 - \frac{Q(s)}{\lambda} & \text{if } s = s'. \end{cases} \quad (5)$$

In the case of uniformization, conditionally on the trajectory $X([0, t_{\max}])$, the virtual jumps form a piecewise homogeneous Poisson process with the intensity constant and equal to $\lambda - Q(X(t_j))$ on every time interval $[t_j, t_{j+1})$ for $j = 0, 1, \dots, |T|$.

The second natural choice is to make virtual jumps distributed as homogeneous Poisson process with intensity $\theta > 0$. Let $R(s) = Q(s) + \theta$. Then the thinning procedure leads to the following probability distribution:

$$\begin{aligned} p(T, V, S) &\propto \nu(s_0) \prod_{k=1}^n P(s_{k-1}, s_k) (\theta + Q(s_{k-1})) \exp \{-(Q(s_{k-1}) + \theta)(\tilde{t}_k - \tilde{t}_{k-1})\} \\ &\quad \times \exp \{-(Q(s_n) + \theta)(t_{\max} - \tilde{t}_n)\} \\ &= \nu(s_0) \prod_{k=1}^n Q(s_{k-1}, s_k)^{1(s_k \neq s_{k-1})} \exp \{-Q(s_{k-1})(\tilde{t}_k - \tilde{t}_{k-1})\} \\ &\quad \times \exp \{-Q(s_n)(t_{\max} - \tilde{t}_n)\} \theta^{|V|} \exp\{-\theta t_{\max}\}, \end{aligned} \quad (6)$$

where the transition matrix P of discrete time Markov chain which generates the skeleton S is given by

$$P(s, s') = \begin{cases} \frac{Q(s, s')}{Q(s) + \theta} & \text{if } s \neq s'; \\ \frac{\theta}{Q(s) + \theta} & \text{if } s = s'. \end{cases} \quad (7)$$

4. Continuous time Bayesian networks

Let $(\mathcal{V}, \mathcal{E})$ denote a directed graph with possible cycles. We write $w \rightarrow u$ instead of $(w, u) \in \mathcal{E}$. For every node $w \in \mathcal{V}$ consider a corresponding space \mathcal{X}_w of possible states. Assume that each space \mathcal{X}_w is discrete. We consider a continuous time stochastic process on the product space $\mathcal{X} = \prod_{w \in \mathcal{V}} \mathcal{X}_w$. Thus a state $s \in \mathcal{X}$ is a configuration $s = (s_w) = (s_w)_{w \in \mathcal{V}}$, where $s_w \in \mathcal{X}_w$. If $\mathcal{W} \subseteq \mathcal{V}$ then we write $s_{\mathcal{W}} = (s_w)_{w \in \mathcal{W}}$ for configuration s restricted to nodes in \mathcal{W} . We also use notation $\mathcal{X}_{\mathcal{W}} = \prod_{w \in \mathcal{W}} \mathcal{X}_w$, so that we can write $s_{\mathcal{W}} \in \mathcal{X}_{\mathcal{W}}$. The set $\mathcal{V} \setminus \{w\}$ will be denoted simply by $-w$. We define the set of parents of node w by

$$\text{pa}(w) = \{u \in \mathcal{V} : u \rightarrow w\},$$

and we define the set of children of node w by

$$\text{ch}(w) = \{u \in \mathcal{V} : w \rightarrow u\}.$$

Suppose we have a family of functions $Q_w : \mathcal{X}_{\text{pa}(w)} \times (\mathcal{X}_w \times \mathcal{X}_w) \rightarrow [0, \infty)$. For fixed $c \in \mathcal{X}_{\text{pa}(w)}$, we consider $Q_w(c; \cdot, \cdot)$ as a conditional intensity matrix (CIM) at node w (only off-diagonal elements of this matrix have to be specified, the diagonal ones are irrelevant). The state of a CTBN at time t is a random element $X(t)$ of the space \mathcal{X} of configurations. Let $X_w(t)$ denote its w th coordinate. The process $\{X_w(t)_{w \in \mathcal{V}}, t \geq 0\}$ is assumed to be Markov and its evolution can be described informally as follows. Transitions at node w depend on the current configuration of the parent nodes. If the state of some parent changes, then node w switches to other transition probabilities. If $s_w \neq s'_w$ then

$$\mathbb{P}(X_w(t + dt) = s'_w | X_{-w}(t) = s_{-w}, X_w(t) = s_w) = Q_w(s_{\text{pa}(w)}, s_w, s'_w) dt.$$

Formally, CTBN is a MPJ with transition intensities given by

$$Q(s, s') = \begin{cases} Q_w(s_{\text{pa}(w)}, s_w, s'_w) & \text{if } s_{-w} = s'_{-w} \text{ and } s_w \neq s'_w \text{ for some } w; \\ 0 & \text{if } s_{-w} \neq s'_{-w} \text{ for all } w, \end{cases}$$

for $s \neq s'$ (of course, $Q(s, s)$ must be defined “by subtraction” to ensure $\sum_{s'} Q(s, s') = 0$).

For a CTBN, the density of sample path $X = X([0, t_{\max}])$ in a bounded time interval $[0, t_{\max}]$ decomposes as follows:

$$p(X) = \nu(X(0)) \prod_{w \in \mathcal{V}} p(X_w \| X_{\text{pa}(w)}) , \quad (8)$$

where ν is the initial distribution on \mathcal{X} and $p(X_w \| X_{\text{pa}(w)})$ is the density of piecewise homogeneous Markov jump process with intensity matrix equal to $Q_w(c; \cdot, \cdot)$ in every time sub-interval such that $X_{\text{pa}(w)} = c$. Formulas for the density of CTBN appear e.g. in (Nodelman et al., 2002b, Sec. 3.1), (Fan et al., 2010, Eq. 2), (Fan and Shelton, 2008, Eq. 1) and (Miasojedow et al., 2014). These formulas give a factorization of the main part of the density as in (8), but in the first three of the cited papers the initial distribution ν is disregarded. There are some subtle problems related to ν , discussed in Miasojedow et al. (2014). Our notation $p(X_w \| X_{\text{pa}(w)})$ is consistent with the notion of “conditioning by intervention”, see e.g. (Lauritzen, 2001). Indeed, $p(X_w \| X_{\text{pa}(w)})$ is the density of the process X_w at node w under the assumption that the sample paths at the parent nodes $X_{\text{pa}(w)}$ are fixed and $X_w(0)$ is given, see e.g. Miasojedow et al. (2014), for details. Below we explicitly write an expression for $p(X_w \| X_{\text{pa}(w)})$ in terms of moments of jumps and the skeleton of the process $(X_w, X_{\text{pa}(w)})$, as in (1). Let $T^w = (t_0^w, \dots, t_i^w, \dots)$ and $T^{\text{pa}(w)} = (t_0^{\text{pa}(w)}, \dots, t_j^{\text{pa}(w)}, \dots)$ denote moments of jumps at node $w \in V$ and at parent nodes, respectively. By convention put $t_0^w = t_0^{\text{pa}(w)} = 0$ and $t_{|T^w|+1}^w = t_{|T^{\text{pa}(w)}|+1}^{\text{pa}(w)} = t_{\max}$. Analogously, S^w and $S^{\text{pa}(w)}$ denote the corresponding skeletons. Thus we divide the time interval $[0, t_{\max}]$ into segments $[t_j^{\text{pa}(w)}, t_{j+1}^{\text{pa}(w)})$, $j = 0, 1, \dots, |T^{\text{pa}(w)}|$ such that $X_{\text{pa}(w)}$ is constant and X_w is homogeneous in each segment. Next we define sets $I_j = \{i > 0 : t_j^{\text{pa}(w)} < t_i^w < t_{j+1}^{\text{pa}(w)}\}$ with notation

$j_{\text{beg}}, j_{\text{end}}$ for the first and the last element of I_j . Analogously to (1), we obtain the following formula:

$$\begin{aligned}
 p(X_w \| X_{\text{pa}(w)}) &= p(T^w, S^w \| S^{\text{pa}(w)}, T^{\text{pa}(w)}) = \prod_{j=0}^{|T^{\text{pa}(w)}|} \left[\prod_{i \in I_j} Q_w(s_j^{\text{pa}(w)}; s_{i-1}^w, s_i^w) \right. \\
 &\times \prod_{i \in I_j \setminus \{j_{\text{beg}}\}} \exp \left\{ -(t_i^w - t_{i-1}^w) Q_w(s_j^{\text{pa}(w)}; s_{i-1}^w) \right\} \\
 &\times \exp \left\{ -(t_{j_{\text{beg}}}^w - t_j^{\text{pa}(w)}) Q_w(s_j^{\text{pa}(w)}; s_{j_{\text{beg}}-1}^w) - (t_{j+1}^{\text{pa}(w)} - t_{j_{\text{end}}}^w) Q_w(s_j^{\text{pa}(w)}; s_{j_{\text{end}}}^w) \right\} \Big].
 \end{aligned} \tag{9}$$

Formula (9) is equivalent to (Nodelman et al., 2002b, Eq. 2) and (Fan and Shelton, 2008, Eq. 1), but expressed in terms of (S, T) .

5. Hidden Markov models

Let $X = \{X(t), 0 \leq t \leq t_{\text{max}}\}$ be a Markov jump process. Suppose that process X cannot be directly observed but we can observe some random quantity Y with probability distribution $L(Y|X([0, t_{\text{max}}]))$. Let us say Y is the evidence and L is the likelihood. We assume that the likelihood depends on X only through the actual sample path $X([0, t_{\text{max}}])$ (does not depend on virtual jumps). The problem is to restore the hidden trajectory of X given Y . From the Bayesian perspective, the goal is to compute/approximate the posterior

$$p(X([0, t_{\text{max}}])|Y) \propto p(X([0, t_{\text{max}}]))L(Y|X([0, t_{\text{max}}])).$$

Function L , transition probabilities Q and initial distribution ν are assumed to be known. To get explicit form of posterior distribution we will consider two typical forms of noisy observation. In the first model, the trajectory of X is observed independently at deterministic time points t_1^*, \dots, t_l^* with corresponding likelihood functions $L_1(y_1|X(t_1^*)), \dots, L_l(y_l|X(t_l^*))$. Since $X(t)$ is constant between jumps, for all $i = 1, \dots, l$ we have $L_i(y_i|X(t_i^*)) = L_i(y_i|s_{i^*})$, where $i^* = \sup\{j : t_j \leq t_i^*\}$. If the trajectory of X is represented by moments of true jumps T , virtual jumps V and skeleton S then the posterior distribution is given by

$$p(T, V, S|Y) \propto p(T, V, S) \prod_{i=1}^l L_i(y_i|s_{i^*}) \tag{10}$$

where $p(T, V, S)$ is given by (3). In Section 3 we considered two scenarios of adding virtual jumps: via uniformization and via a homogeneous Poisson process. The corresponding densities are given by (4) and (6), respectively. Observe that for both variants of adding virtual jumps, the posterior can be expressed in the following form:

$$p(S|T, V, Y) = \nu(s_0)g_0(s_0) \prod_{k=1}^n P(s_{k-1}, s_k)g_k(s_k), \tag{11}$$

where P is a Markov transition matrix, g_k are some functions which can depend on T, V, Y and $n = |T| + |V|$. Hence the skeleton, conditionally on all the jumps and the evidence, can be treated as a hidden Markov model with discrete time.

In the second typical model of observation, the evidence Y is a fully observed continuous time stochastic process depending on X . Expressly, we assume that Y , given the trajectory of X , is a piecewise homogeneous Markov jump process such that the pair (X, Y) is a CTBN with the graph structure $X \rightarrow Y$. Thus the likelihood can be expressed by (9), with $X^w = Y$ and $X^{\text{pa}(w)} = X$. In this case we can easily obtain the same conclusion as before: for both variants of adding virtual jumps, the posterior can be expressed in the form (11). The skeleton is conditionally a hidden Markov model.

6. MCMC algorithm

Let us recall the standing assumption that evidence Y depends only on the trajectory of X but not on virtual jumps. This assumption covers most of usual scenarios and clearly implies that $p(V|T, S, Y) = p(V|T, S)$. Now we are able to state the main algorithm. Similarly to (Rao and Teh, 2012, 2013), a single step of the iterative procedure is the following. We take the trajectory obtained in the previous step, represented by (T, S) (ignoring the virtual jumps). First we sample a new set of virtual jumps V . We can use two variants of sampling. In the case of uniformization, V is a piecewise homogeneous Poisson process with intensities $\lambda - Q(X(t_k))$. Alternatively, V is a homogeneous Poisson process with intensity θ . Next we generate a new skeleton S' using Markov kernel with $p(S|T, V, Y)$ as invariant distribution. In (Rao and Teh, 2012, 2013), the authors use independent sampling by the forward filtering – backward sampling (FFBS) algorithm. In our approach we propose to use the particle Gibbs algorithm invented by Andrieu et al. (2010), which is described in the next section. Note that a new skeleton with fixed times of potential jumps leads to a new allocation of true and virtual jumps. So we obtain a new trajectory described by (T', V', S') such that $T \cup V = T' \cup V'$. Finally we remove virtual jumps to obtain a new state (T', S') . The algorithm is summarized below.

Algorithm 2 Single step of MCMC algorithm.

Input: Previous state (T, S) and observation Y .

1. Add virtual jumps V .

2. Draw new skeleton S' from Markov kernel $K(S, \cdot)$ targeting $p(S|T, V, Y)$. Skeleton S' defines new allocation of virtual and true jumps T', V' such $T \cup V = T' \cup V'$.

3. Remove virtual jumps V' .

return new state (T', S') .

The next proposition shows that this algorithm is ergodic.

Proposition 3 *Assume that $\lambda > \max_s Q(s)$ in the case of uniformization or $\theta > 0$ in the case of homogeneous virtual jumps. Assume that kernel $K(S, S')$ leaves distribution $p(S|T, V, Y)$ invariant and $K(S, S') > 0$ for all S' such that $p(S'|T, V, Y) > 0$. Then MCMC algorithm described above is ϕ -irreducible, aperiodic with stationary distribution $\pi(T, S) = p(T, S|Y)$. Thus for π -almost all initial positions, the algorithm is ergodic, i.e.*

$$\|M((T, S), \cdot)^m - \pi(\cdot)\|_{\text{tv}} \xrightarrow{m \rightarrow \infty} 0,$$

where M denotes the kernel of our MCMC algorithm.

Proof By construction it is clear that $p(T, S|Y)$ is a stationary distribution. Since $K(S, S) > 0$, with positive probability it happens that the skeleton does not change and hence virtual and true jumps remain unchanged. Clearly $M((T, S), (T, S)) > 0$ and so Markov chain M is aperiodic. The assumption $\lambda > \max_s Q(s)$ or $\theta > 0$ ensures that the step of adding virtual jumps can reach any configuration of virtual jumps. Together with the assumption $K(S, S') > 0$ it leads to the conclusion that all states (T, S) in the support of π are reachable. Hence M is ϕ -irreducible. It is now enough to invoke the well-known fact that ϕ -irreducibility and aperiodicity imply ergodicity in total variation norm, see for example (Theorem 4, [Roberts and Rosenthal, 2004](#)). ■

Remark 4 Condition $K(S, S') > 0$ is clearly satisfied by FFBS algorithm, because it is equivalent to independent sampling from $p(S|T, V, Y)$. This condition is also satisfied by the particle Gibbs algorithm described in the next section.

In the case of CTBN we can use its dependence structure by introducing a Gibbs sampler over nodes of the graph $(\mathcal{V}, \mathcal{E})$. The same idea was exploited in ([El-Hay et al., 2008](#); [Rao and Teh, 2013](#)). By (8), the full conditional distribution of node w given rest of the graph has the density

$$p(X_w|X_{-w}, Y) \propto \nu(X_w(0)|X_{-w}(0))p(X_w||X_{\text{pa}(w)}) \prod_{u \in \text{ch}(w)} p(X_u||X_{\text{pa}(u)})L(Y|X). \quad (12)$$

The density $p(X_w||X_{\text{pa}(w)})$ corresponds to a piecewise homogeneous Markov process. The expression $\prod_{u \in \text{ch}(w)} p(X_u||X_{\text{pa}(u)})$ can be treated as a part of likelihood, similarly as $L(Y|X)$. Note that the conditional initial distribution $\nu(X_w(0)|X_{-w}(0))$ may be replaced in formula (12) by the joint initial distribution $\nu(X(0))$, because these two quantities are proportional as functions of $X_w(0)$. The step which leaves $p(X_w|X_{-w}, Y)$ as invariant measure can be realized by the general algorithm described above. The Gibbs sampler for CTBN is summarized below.

Algorithm 3 Gibbs sampler for CTBN.

for $w \in \mathcal{V}$ (in a deterministic or random order) **do**
 Simulate (T_w, S_w) using single step of MCMC algorithm targeting $p(X_w|X_{-w}, Y)$, with X_{-w} fixed.
end for

Note that if observations of different nodes are independent i.e. $L(Y|X) = \prod_{w \in \mathcal{V}} L_w(Y|X_w)$ then the full conditional distributions defined by (12) reduce to

$$p(X_w|X_{-w}, Y) \propto \nu(X(0))p(X_w||X_{\text{pa}(w)})L_w(Y|X_w) \prod_{u \in \text{ch}(w)} p(X_u||X_{\text{pa}(u)}).$$

Hence within the Gibbs sampler, the step for node w need not involve evaluation of full likelihood. An immediate corollary from Proposition 3 is that the Gibbs sampler for CTBN is also ergodic. Note that in the step of adding virtual jumps we can choose the intensity parameters λ or θ globally but, more generally, we can define different intensities for every node w , say λ_w or θ_w .

Corollary 5 *Assume that for every node $w \in \mathcal{V}$ we have $\lambda_w > \max_{s^w, s^{\text{pa}(w)}} Q_v(s^{\text{pa}(w)}; s^w)$ in the case of uniformization or $\theta_w > 0$ for homogeneous virtual jumps. Then the Gibbs sampler for CTBN (with either the particle Gibbs or FFBS used in sampling of a new skeleton) is ergodic.*

7. Particle Gibbs

In this section we will use notation which is standard in the literature on sequential Monte Carlo (SMC). Let $s_{0:k} = (s_0, s_1, \dots, s_k)$. Consider a sequence of unnormalized densities $\gamma_k(s_{0:k})$ on increasing product spaces \mathcal{S}^{k+1} for $k = 1, \dots, n$. The corresponding normalized probability densities are

$$\pi_k(s_{0:k}) = \frac{\gamma_k(s_{0:k})}{Z_k},$$

where $Z_k = \int \gamma_k(s_{0:k}) ds_{0:k}$ is the normalizing constant. A special case is the so-called state-space model where $\pi_k(s_{0:k}) = p(s_{0:k} | y_{0:k})$ and $\gamma_k(s_{0:k}) = p(s_{0:k}, y_{0:k})$. In the sequel we consider the state-space model because the distribution of skeleton S given times of jumps (true and virtual; (T, V)) fits in this scheme c.f. (11).

Particle MCMC methods introduced by [Andrieu et al. \(2010\)](#) provide a general framework for constructing an MCMC kernel targeting $\pi(s_{0:k})$ with transition rule based on SMC algorithms. Before we describe particle MCMC in detail, we first have to recall standard SMC methods, e.g. [Doucet et al. \(2001\)](#); [Doucet and Johansen \(2009\)](#); [Del Moral et al. \(2006\)](#); [Pitt and Shephard \(1999\)](#). SMC sequentially approximates each of probability distributions π_k by an empirical distribution

$$\hat{\pi}_k^N(ds_{0:k}) = \sum_{i=1}^N \frac{w_k^i}{\sum_j w_k^j} \delta_{\xi_{0:k}^i}(ds_{0:k}), \quad (13)$$

where $\{\xi_{0:k}^i, w_k^i\}_{i=1}^N$ is a weighted particle system. The system is propagated as follows. Given previous approximation $\{\xi_{0:k-1}^i, w_{k-1}^i\}_{i=1}^N$ at time $k-1$, first we draw $\{\tilde{\xi}_{0:k-1}^i\}_{i=1}^N$ from the multinomial distribution with probabilities proportional to weights $\{w_{k-1}^i\}_{i=1}^N$ (this resampling step is introduced to avoid degeneracy of weights). Next we generate new particles $\{\xi_k^i\}_{i=1}^N$ according to $r_k(\cdot | \tilde{\xi}_{0:k-1}^i)$ and set $\xi_{0:k}^i = (\tilde{\xi}_{0:k-1}^i, \xi_k^i)$. Here r_k is an instrumental kernel from \mathcal{S}^k to \mathcal{S} (identified with a conditional density). Finally we compute new weights

$$w_k^i = \frac{\gamma_k(\xi_{0:k}^i)}{\gamma_{k-1}(\tilde{\xi}_{0:k-1}^i) r_k(\xi_k^i | \tilde{\xi}_{0:k-1}^i)}. \quad (14)$$

The SMC algorithm is summarized below.

Algorithm 4 SMC algorithm.

Initialization

for $i = 1, \dots, N$ **do**

 Draw $\xi_0^i \sim r_0(\cdot)$.

Compute weights

$$w_0^i = \frac{\gamma_0(\xi_0^i)}{r_0(\xi_0^i)} .$$

end for

Main loop

for $k = 1, \dots, n$ **do**
Resampling step:

Draw ancestors

$$\{a_k^i\}_{i=1}^N \sim \text{Multinomial} \left(N, \frac{w_{k-1}^1}{\sum_j w_{k-1}^j}, \dots, \frac{w_{k-1}^N}{\sum_j w_{k-1}^j} \right)$$

and set

$$\tilde{\xi}_{0:k-1}^i = \xi_{0:k-1}^{a_k^i} .$$

Propagation step:
for $i = 1, \dots, N$ **do**

Draw

$$\xi_k^i \sim r_k(\cdot | \tilde{\xi}_{0:k-1}^i)$$

and set

$$\xi_{0:k}^i = (\tilde{\xi}_{0:k-1}^i, \xi_k^i) .$$

 Compute weights w_k^i from (14).

end for
end for

The particle Gibbs (PGS) algorithm introduced by [Andrieu et al. \(2010\)](#) is based on SMC algorithm. Given a previous path $s_{0:n}$ we run an SMC algorithm with one path fixed, say $\xi_{0:n}^N = s_{0:n}$, and obtain system of particles $\{\xi_{0:n}^i, w_n^i\}_{i=1}^N$. Next we draw a new path $s'_{0:n}$ with probability

$$\mathbb{P}(s'_{0:n} = \xi_{0:n}^i) \propto w_n^i .$$

This procedure defines the following Markov kernel:

$$K(s_{0:n}, \cdot) = \mathbb{E}(\hat{\pi}_n^N(\cdot) | s_{k:n} = \xi_{0:n}^N) ,$$

where $\hat{\pi}_n^N$ is defined by (13). It is shown that for every number of particles N larger than one, π_n is a stationary distribution for kernel K ([Andrieu et al., 2010](#)).

There is a well-known problem of path degeneracy of SMC samplers ([Doucet and Johansen, 2009](#)). For large n , the beginning of the path $s'_{0:n}$ can be based on only few

trajectories. This may lead to poor mixing of the particle Gibbs sampler, because kernel K with high probability leaves the beginning of trajectory unchanged, see (Chopin and Singh, to appear 2015; Lindsten and Schön, 2013; Lindsten et al., 2014b). A remedy for this problem can be an additional step of ancestor resampling proposed by Lindsten et al. (2014b). Let $\xi_{0:n}^N = s_{0:n}$ be the fixed trajectory in the particle Gibbs algorithm. For every $k = 1, \dots, n$ we sample an ancestor of ξ_k^N from the set of trajectories $\{\xi_{0:k-1}^i\}_{i=1}^N$ with probabilities proportional to weights

$$w_{k-1|n}^i = w_{k-1}^i \frac{\gamma_n((\xi_{0:k-1}^i, s_{k:n}))}{\gamma_k(\xi_{0:k-1}^i)}. \quad (15)$$

The above modification does not change the invariant measure of the Markov kernel (Theorem 1, Lindsten et al., 2014b).

Now we are ready to describe precisely the step of sampling a new skeleton in the MCMC algorithm for hidden Markov jump processes. Let us recall (11). The conditional distribution of skeleton given moments of jumps and evidence is of the form

$$p(S|T, V, Y) = \nu(s_0)g_0(s_0) \prod_{k=1}^n P(s_{k-1}, s_k)g_k(s_k),$$

where P is a transition matrix of some Markov chain. A standard choice is to use priors as instrumental kernels r i.e. $r_0 = \nu$ and $r_k = P$ for $k > 0$. This choice leads to a simplified form of weights:

$$w_0^i = g_0(\xi_0^i), \quad w_k^i = g_k(\xi_k^i), \quad w_{k-1|n}^i = w_{k-1}^i P(\xi_{k-1}^i, s_k),$$

for $i = 1, \dots, N$ and $k = 1, \dots, n$. If the algorithm is applied in a CTBN setting within a Gibbs sampler step, then the initial conditional distribution $\nu(X_w(0)|X_{-w}(0))$ in (12) might be difficult to sample from. Then we can use a different instrumental distribution r_0 and compute weights $w_0^i = g_0(\xi_0^i)\nu(\xi_0^i)/r_0(\xi_0^i)$. However, in many scenarios the initial configuration is deterministic (ν is concentrated at a single configuration) and then this problem disappears. In algorithm 5 we summarize the particle Gibbs with ancestor sampling (PGAS).

Remark 6 *There exists also a Metropolis type of particle MCMC algorithm (PMH). However, straightforward application of PMH within MCMC algorithm for Markov jump processes is not possible, because the acceptance probability in PMH depends on estimates of the norming constant Z_n obtained in two consecutive steps. In the case of Markov jump processes, the dimension of the state space $(n+1)$ can be different in every step, because virtual jumps are either added or removed. Therefore to implement PMH in this context, transdimensional Metropolis type moves would be needed.*

Algorithm 5 PGAS for sampling the new skeleton.

Input: Current skeleton $s_{0:n}$.

Output: New skeleton $s'_{0:n}$.

 Set $\xi_{0:n}^N = s_{0:n}$.

 Compute weights $w_k^N = g_k(s_k)$ for $k = 0, \dots, n$.

for $i = 1, \dots, N - 1$ **do**

 Draw $\xi_0^i \sim \nu(\cdot)$.

 Compute corresponding weights $w_0^i = g_0(\xi_0^i)$.

end for
for $k = 1, \dots, n$ **do**

 (**Resampling**)

Draw ancestors

$$\{a_k^i\}_{i=1}^N \sim \text{Multinomial} \left(N, \frac{w_{k-1}^1}{\sum_j w_{k-1}^j}, \dots, \frac{w_{k-1}^N}{\sum_j w_{k-1}^j} \right)$$

and set

$$\tilde{\xi}_{0:k-1}^i = \xi_{0:k-1}^{a_k^i}.$$

 (**Ancestor resampling**)

 Draw J with probability

$$\mathbb{P}(J = i) = \frac{w_{k-1}^i P(\tilde{\xi}_{k-1}^i, s_k)}{\sum_{j=1}^N w_{k-1}^j P(\tilde{\xi}_{k-1}^j, s_k)},$$

and set

$$\xi_{0:n}^N = (\tilde{\xi}_{0:k-1}^J, s_{k:n}).$$

 (**Propagation step**)

for $i = 1, \dots, N - 1$ **do**

 Draw $\xi_k^i \sim P(\tilde{\xi}_{k-1}^i, \cdot)$ and set $\xi_{0:k}^i = (\tilde{\xi}_{0:k-1}^i, \xi_k^i)$.

 Compute weights $w_i^k = g_k(\xi_k^i)$.

end for
end for

 Draw I with probability

$$\mathbb{P}(I = i) = \frac{w_n^i}{\sum_{j=1}^N w_n^j},$$

and set

$$s'_{0:n} = \xi_{0:n}^I.$$

8. Numerical experiments

In this section we present results of simulations which demonstrate the efficiency of the proposed algorithm. We concentrate on the case of CTBNs. We start with a toy example with only two nodes joined by a single arrow ($X \rightarrow Y$), i.e. the simplest hidden Markov model with continuous time. Next we consider a CTBN with a chain structure. The last example is the Lotka-Volterra predator-prey model. To the best of our knowledge the algorithm of [Rao and Teh \(2013\)](#) is the most efficient method to deal with hidden continuous time Markov processes. For this reason we use it in our comparisons. In the simulations we use RCPP ([Eddelbuettel et al., 2011](#)) implementation of algorithms. All our codes are available at a request.

8.1 Toy example

Consider a CTBN presented in [Figure 1](#). Both nodes (X, Y) have two possible states, $\{1, 2\}$. Node Y is fully observed, X is hidden (apart from the beginning and the end of its trajectory). The transition intensities of X are independent of Y and given by

$$Q_X = \begin{pmatrix} -10 & 10 \\ 10 & -10 \end{pmatrix}.$$

The conditional intensities of Y are given by

$$Q_{Y|X=1} = \begin{pmatrix} -10 & 10 \\ 10 & -10 \end{pmatrix}, \quad Q_{Y|X=2} = \begin{pmatrix} -100 & 100 \\ 100 & -100 \end{pmatrix}.$$

Assume that we observe the full trajectory of Y in time interval $[0, 1]$ and we also observe X at moments 0 and 1. Our goal is to sample from the posterior distribution of $X([0, 1])$ given the evidence $(X(0), X(1), Y([0, 1]))$. We run our particle MCMC algorithm in two versions. Times of virtual jumps are added via uniformization with $\lambda = 20$ in the first version and distributed according to the homogeneous Poisson process with intensity $\theta = 10$ in the second version. In both cases the expected number of virtual jumps is approximately the same. We run our algorithm with PGAS with 2 and 4 particles. For a comparison we apply [Rao and Teh \(2013\)](#) algorithm with FFBS subroutine. The actual (hidden) trajectory $X([0, 1])$, the posterior means and standard deviation of MCMC approximation are presented in [Figure 2](#). These results are based on 100 replications, each MCMC run of length 1000 with a burn-in time 100. We observe that the estimated trajectory is very similar for all the compared methods. The difference is in the variance and consequently in the width of the “confidence bands”. In this example the algorithm with uniformization outperforms the algorithm with homogeneous virtual jumps. As it was expected, our algorithms with PGAS have the variance greater than those with FFBS but the difference between PGAS with 4 particles and FFBS is not too big. To analyze the rate of convergence of the algorithms we also compute standard deviation of sufficient statistics (number of jumps and time spent in each state). The results are shown in [Figure 3](#). Again we obtain similar conclusions. It is clearly visible that the algorithms with FFBS have lower variance than those with PGAS. However, the difference is rapidly decreasing with increasing number of particles. Approximately the cost of FFBS is of order $\mathcal{O}(|\mathcal{X}|^2 \mathbb{E}(n))$ and the cost of PGAS

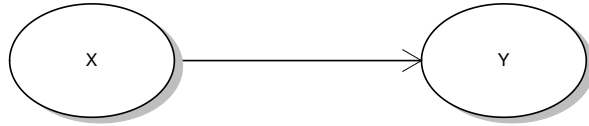


Figure 1: Toy example

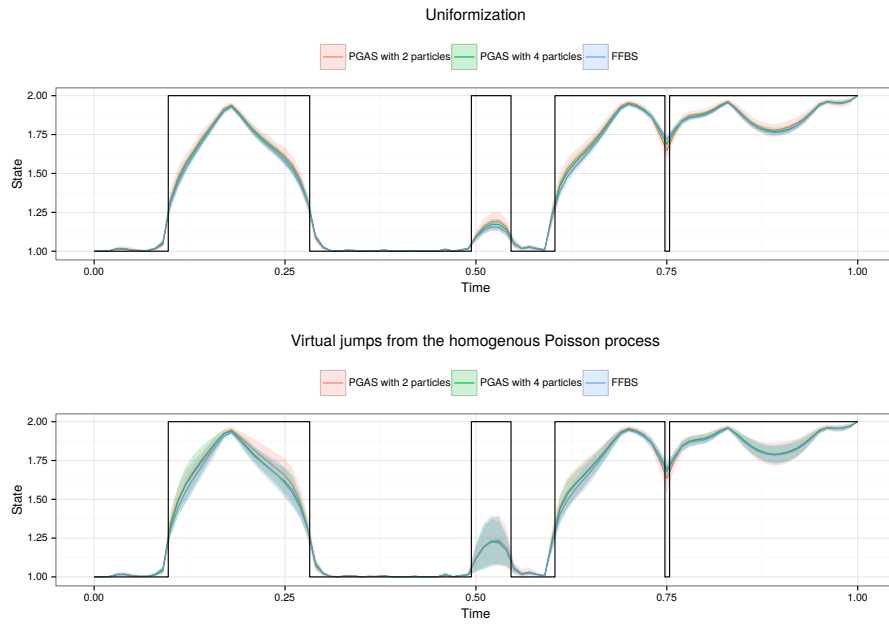


Figure 2: Posterior mean and standard deviation of MCMC approximation for the toy example.

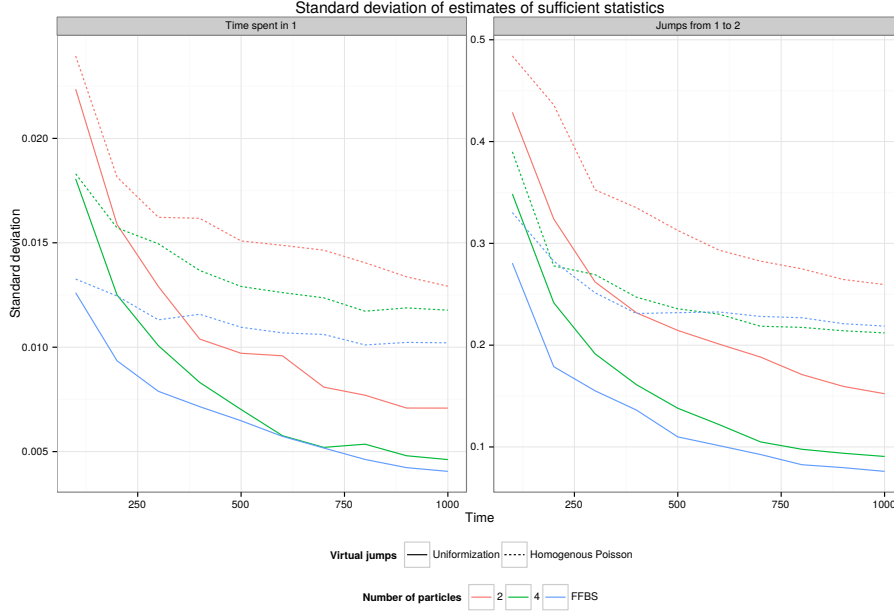


Figure 3: Standard deviation of sufficient statistics versus number of iterations of MCMC algorithm.

is $\mathcal{O}(NE(n))$. Hence in the example under consideration, we obtain comparable quality of estimation for the algorithms with FFBS and PGAS which have the same computational cost.

8.2 Chain network

The model considered in this subsection is based on (Rao and Teh, 2013, Subsection 5.5). It is a network which consists of M nodes equipped with the “chain” graph $1 \rightarrow 2 \rightarrow \dots \rightarrow M$. For each node the set of possible states is $\{1, \dots, S\}$. The transition intensities of every node, except the first one, depend on current state of the previous node. Namely, off-diagonal elements of intensity matrix are given by

$$Q_1(x_1, x'_1) = \begin{cases} \frac{1}{2} & \text{if } x'_1 = (x_1 + 1) \bmod S ; \\ \frac{1}{2(S-2)} & \text{otherwise ,} \end{cases}$$

$$Q_m(x_{m-1}; x_m, x'_m) = \begin{cases} \frac{1}{S-1} & \text{if } x_m = x_{m-1} , \\ 1 & \text{if } x_{m-1} \neq x_m \text{ and } x'_m = x_{m-1} ; \\ \frac{1}{S-2} & \text{otherwise.} \end{cases}$$

In words, the head node leaves the current state x_1 with intensity 1 and prefers to jump to $x_1 + 1 \bmod S$. For any other node, if its current state x_m differs from that of parent state x_{m-1} then intensity of jumping is 2 and the process prefers to jump to x_{m-1} . If $x_m = x_{m-1}$

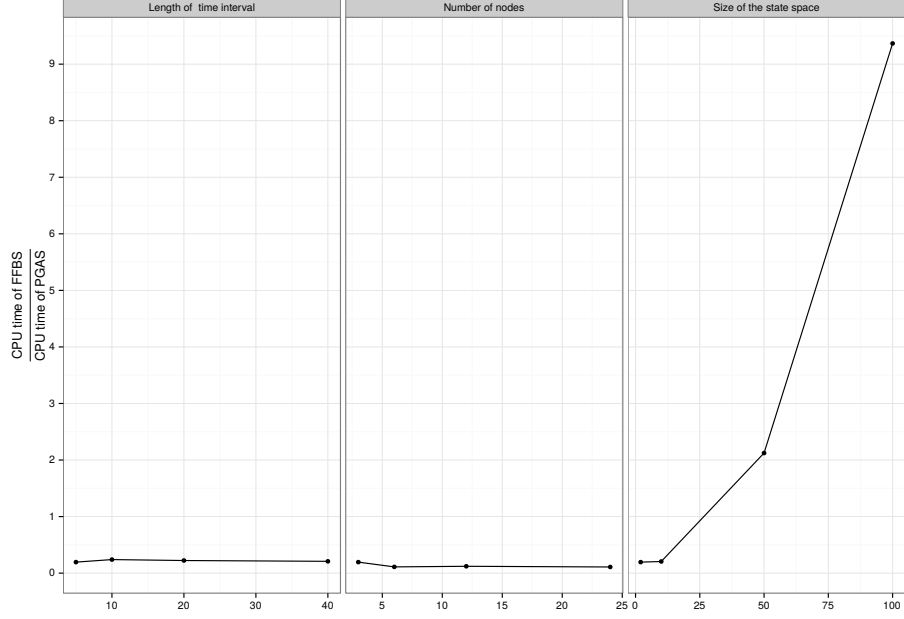


Figure 4: Ratio of CPU time (FFBS/PGAS) needed to generate a sample with $ESS = 100$ for the chain model: with increasing length of time interval (left), increasing number of nodes (center), increasing size of the state space (right).

then the process leaves the current state with intensity 1 and chooses a new state at random. We observe the process at the beginning and at the end of time interval $[0, T]$.

We run two MCMC algorithms targeting the posterior distribution over latent trajectories. Our algorithm with PGAS based on $N = 10$ particles is compared with Rao and Teh’s algorithm with FFBS. Both the algorithms use uniformization, with λ equal to twice the intensity of leaving the current state.

We begin with a network with $M = 3$ nodes, $S = 2$ states of every node and time length $T = 5$. In the first series of our experiments, we increase the number of nodes to $M = 6, 12, 24$. In the second series we increase the length of time interval to $T = 10, 20, 40$. Finally we change the size of the state space to $S = 10, 50, 100$. For each of the scenarios we run 20 replications of each of the MCMC algorithms. We use CODA R package (Plummer et al., 2006) to estimate the effective sample size of the following statistics: time spent in each state and number of jumps, for nodes $m = 1, \dots, M$. The median of all these quantities serves as an estimator of ESS for the whole CTBN. In Figure 4 we present the ratio of CPU time needed to generate a sample with ESS equal to 100. In the beginning, when state space is of size $S = 2$, FFBS is more effective, around 4-5 times as fast as PGAS. This is what should be expected, since the cost of PGAS with 10 particles is higher than the cost of FFBS for a small space size. Also as expected, the ratio does not change significantly if the number of nodes increases. The same seems to happen if we increase the length of time interval. This last fact is slightly surprising, because the rate of convergence of particle

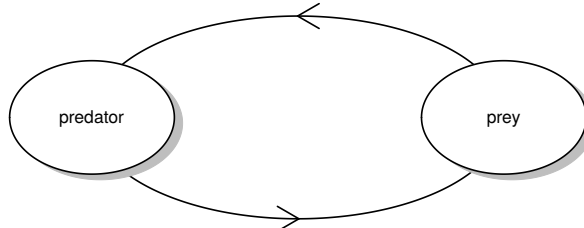


Figure 5: Predator-Prey model as CTBN.

Gibbs depends on the number of jumps, see [Andrieu et al. \(2013\)](#); [Lindsten et al. \(2014a\)](#). However, if the size of the state space increases then the cost of our proposed method becomes significantly lower than that of [Rao and Teh \(2013\)](#) approach. For instance if $S = 50$ then our algorithm is twice as fast as [Rao and Teh \(2013\)](#) and for $S = 100$ it is more than 9 times as fast. Experiments with a different number of particles ($N = 5, 20, 50$) lead to the same conclusions.

8.3 Lotka-Volterra model

The last example is Lotka-Volterra model ([Wilkinson, 2009](#); [Oppen and Sanguinetti, 2008](#)), which describes evolution of two interacting populations of prey and predator species. This process can be viewed as the two node CTBN shown in Figure 5. Let x and y represent the size of prey and predator populations, respectively. The transition intensities are given by

$$\begin{aligned} Q(\{x, y\}, \{x + 1, y\}) &= \alpha x, & Q(\{x, y\}, \{x - 1, y\}) &= \beta xy, \\ Q(\{x, y\}, \{x, y + 1\}) &= \delta xy, & Q(\{x, y\}, \{x, y - 1\}) &= \gamma y, \end{aligned}$$

All other intensities are 0. The state space is infinite: $\{0, 1, \dots\} \times \{0, 1, \dots\}$. Following [Rao and Teh \(2013\)](#) we set the parameters as follows: $\alpha = \gamma = 5 \times 10^{-4}$ and $\beta = \delta = 1 \times 10^{-4}$. We consider the process in time interval $[0, 3000]$ with known initial position and noisy observations $Y(t)$ at discrete times uniformly spaced in interval $[0, 1500]$ with the likelihood given by

$$p(Y(t)|X(t)) \propto \left[2^{|X(t)-Y(t)|} + 10^{-6} \right]^{-1}.$$

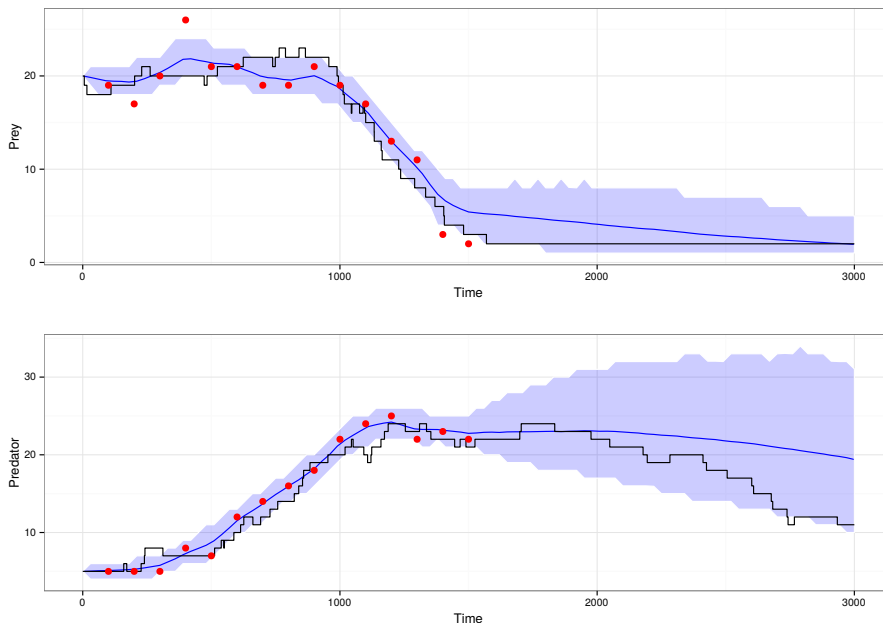


Figure 6: Results of MCMC approximation for Predator-Prey model. Black line - true path, blue line - posterior mean, shadow - 90% credible interval, red points - observations.

Given this evidence, we estimate the posterior distribution over sample paths of X using our sampler. We compute estimates of X in the interval $[0, 1500]$ (where observations Y are available) and also predict values of X in the interval $[1500, 3000]$. Due to unboundedness of intensities we are not able to use uniformization technique and so we use homogenous virtual jumps with $\theta = 30$. For this choice of θ , the number of virtual jumps is approximately equal to the number of true jumps. We run MCMC simulation of length 1000 with 100 initial iterations treated as burn-in time. In the PGAS we use 100 particles. The results of simulation are given in Figure 6. We conclude that the quality of estimates is the same as in Rao and Teh (2013). Note that in opposition to Rao and Teh’s algorithm we need not truncate the state space and the computational cost of our algorithm is significantly lower, namely $\mathcal{O}(100\mathbb{E}(n))$ for our method and $\mathcal{O}(200^2\mathbb{E}(n))$ for Rao and Teh, with state space truncated to $\{0, \dots, 200\} \times \{0, \dots, 200\}$ as suggested by these authors.

9. Conclusions

In the present paper we propose a new MCMC algorithm for sampling from the posterior distribution of hidden trajectory of a Markov jump process. The general idea is the same as in Rao and Teh (2013), namely we alternately add virtual jumps and update the skeleton of the process. The main novelty is that our algorithm uses PGAS to sample the skeleton, while Rao and Teh use FFBS. Thus instead of sampling exactly from a conditional distribution, we make a step of a Markov chain which preserves this distribution. This modification

has some disadvantages, as slower mixing of the entire procedure. However, there are also important advantages of our approach. Unlike previous methods our algorithm can be implemented even if the state space is infinite. The cost of a single step of the proposed algorithm does not depend on the size of the state space. Consequently, we can recommend our algorithm for problems where the space is either infinite or finite but large. If the size of the state space is not big, then our algorithm with PGAS converges slightly slower than the algorithm with FFBS. However, the difference between them is rapidly diminishing if we increase the number of particles in PGAS.

In the present paper we describe an algorithm for time homogeneous processes, only to avoid too many technical details. However, the generalization to non-homogeneous processes is rather straightforward. For details of adding virtual jumps in non-homogeneous case we refer to [Rao and Teh \(2012\)](#). Since PGAS can easily deal with general state spaces, our algorithm can also be applied to piecewise deterministic Markov jump processes on general state spaces (i.e. processes which evolve deterministically between jumps and move according to some Markov kernel at moments of jumps).

We note that an important issue is to find an optimal number of virtual jumps. Small number of virtual jumps can lead to poor mixing of moments of jumps. In the case of PGAS, large number of virtual jumps not only increases computational cost, as it is in the case of FFBS, but may have a negative impact on mixing of the whole algorithm. It is because the convergence rate of particle Gibbs depends on the length of simulated trajectory.

Appendix A.

Proof [Proof of Proposition 1] We are to check that if (\tilde{T}, \tilde{S}) has the distribution given by (3) then $(\tilde{T}_J, \tilde{S}_J)$ is distributed according to (1). First we compute the distribution of waiting time for the next true jump. Without loss of generality we can assume that the previous jump occurred at time 0 and $X(0) = s$. To get the first true jump we generate subsequent moments of potential jumps $\tilde{t}_1, \dots, \tilde{t}_i \dots$ such that $\tilde{t}_i - \tilde{t}_{i-1}$ are i.i.d. from $Exp(R(s))$. The candidate is accepted with probability $Q(s)/R(s)$. Hence

$$\begin{aligned}
 \mathbb{P}(t_1 \leq t) &= \sum_{k=1}^{\infty} \mathbb{P}(\tilde{t}_k < t) \left(1 - \frac{Q(s)}{R(s)}\right)^{k-1} \frac{Q(s)}{R(s)} \\
 &= \sum_{k=1}^{\infty} \mathbb{P}\left(\sum_{i=1}^k (\tilde{t}_i - \tilde{t}_{i-1}) < t\right) \left(1 - \frac{Q(s)}{R(s)}\right)^{k-1} \frac{Q(s)}{R(s)} \\
 &= \sum_{k=1}^{\infty} \int_0^t \frac{R(s)^k}{(k-1)!} u^{k-1} \exp\{-R(s)u\} du \left(1 - \frac{Q(s)}{R(s)}\right)^{k-1} \frac{Q(s)}{R(s)} \\
 &= \int_0^t \sum_{k=1}^{\infty} \frac{(R(s) - Q(s))^{k-1} u^{k-1}}{(k-1)!} \exp\{-R(s)u\} du Q(s) \\
 &= \int_0^t \exp\{[R(s) - Q(s)]u - R(s)u\} du Q(s) \\
 &= \int_0^t Q(s) \exp\{-Q(s)u\} du = 1 - \exp\{-Q(s)t\}.
 \end{aligned}$$

We have obtained an expression which is exactly the c.d.f. of waiting time for the next jump of process with intensity matrix Q . To conclude the proof, it is enough to note that

$$\begin{aligned}\mathbb{P}(s_1 = s' | s_0 = s) &= \mathbb{P}(\tilde{s}_1 = \tilde{s}' | \tilde{s}_0 = \tilde{s}, \tilde{s}_0 \neq \tilde{s}_1) \\ &= \frac{Q(\tilde{s}, \tilde{s}')/R(\tilde{s})}{Q(\tilde{s})/R(\tilde{s})} = \frac{Q(\tilde{s}, \tilde{s}')}{Q(\tilde{s})} .\end{aligned}$$

■

Proof [Proof of Corollary 2] By construction of the thinning procedure and by Proposition 1 we have

$$\begin{aligned}p(V_j | X(t_{j-1}) = s, t_{j-1}, t_j) &= \frac{p(V_j, t_{j-1}, t_j | s)}{p(t_{j-1}, t_j | s)} \\ &= \frac{R(s)^{|V_j|+1} \frac{(R(s) - Q(s))^{|V_j|} Q(s)}{R(s)^{|V_j|+1}} \exp\{-(t_j - t_{j-1})R(s)\}}{Q(s) \exp\{-(t_j - t_{j-1})Q(s)\}} \\ &= (R(s) - Q(s))^{|V_j|} \exp\{-(t_j - t_{j-1})(R(s) - Q(s))\} .\end{aligned}$$

■

References

- Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.
- Christophe Andrieu, Anthony Lee, and Matti Vihola. Uniform ergodicity of the iterated conditional smc and geometric ergodicity of particle gibbs samplers. *arXiv preprint arXiv:1312.6432*, 2013.
- Richard J Boys, Darren J Wilkinson, and Thomas BL Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18(2):125–135, 2008.
- Nicolas Chopin and Sumeetpal S Singh. On the particle gibbs sampler. *Bernoulli*, to appear 2015.
- Erhan Cinlar. *Introduction to stochastic processes*. Prentice-Hall, 1975. ISBN 0134980891.
- Ido Cohn, Tal El-Hay, Nir Friedman, and Raz Kupferman. Mean field variational approximation for continuous-time bayesian networks. *The Journal of Machine Learning Research*, 11:2745–2783, 2010.
- Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

- Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12:656–704, 2009.
- Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pages 3–14. Springer, 2001.
- Dirk Eddelbuettel, Romain François, J Allaire, John Chambers, Douglas Bates, and Kevin Ushey. Rcpp: Seamless r and c++ integration. *Journal of Statistical Software*, 40(8): 1–18, 2011.
- Tal El-Hay, Nil Friedman, and Raz Kupferman. Gibbs sampling in factorized continuous-time markov processes. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 169–178, Corvallis, Oregon, 2008. AUAI Press.
- Yu Fan and Christian R. Shelton. Sampling for approximate inference in continuous time Bayesian networks. In *Tenth International Symposium on Artificial Intelligence and Mathematics*, 2008.
- Yu Fan, Jing Xu, and Christian R. Shelton. Importance sampling for continuous time Bayesian networks. *Journal of Machine Learning Research*, 11(Aug):2115–2140, 2010.
- Paul Fearnhead and Chris Sherlock. An exact gibbs sampler for the markov-modulated poisson process. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(5):767–784, 2006. ISSN 1467-9868. doi: 10.1111/j.1467-9868.2006.00566.x.
- Asger Hobolth and Eric A. Stone. Simulation from endpoint-conditioned, continuous-time markov chains on a finite state space, with applications to molecular evolution. *The Annals of Applied Statistics*, 3(3):1204–1231, 2009.
- Arne Jensen. Markoff chains as an aid in the study of markoff processes. *Scandinavian Actuarial Journal*, 1953(sup1):87–91, 1953.
- Steffen L Lauritzen. Causal inference from graphical models. *Complex stochastic systems*, pages 63–107, 2001.
- P. A. W. Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes with degree-two exponential polynomial rate function. *Operations Research*, 27(5):pp. 1026–1040, 1979.
- Fredrik Lindsten and Thomas B Schön. Backward simulation methods for monte carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.
- Fredrik Lindsten, Thomas Schön, and Michael I Jordan. Ancestor sampling for particle gibbs. In *Advances in Neural Information Processing Systems*, pages 2591–2599, 2012.
- Fredrik Lindsten, Randal Douc, and Eric Moulines. Uniform ergodicity of the particle gibbs sampler. *arXiv preprint arXiv:1401.0683*, 2014a.

- Fredrik Lindsten, Michael I Jordan, and Thomas B Schön. Particle gibbs with ancestor sampling. *The Journal of Machine Learning Research*, 15(1):2145–2184, 2014b.
- Blazej Miasojedow, Wojciech Niemiro, John Noble, and Krzysztof Opalski. Metropolis-type algorithms for continuous time bayesian networks. *arXiv preprint arXiv:1403.4035*, 2014.
- Uri Nodelman, Christian R Shelton, and Daphne Koller. Continuous time bayesian networks. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 378–387, 2002a.
- Uri Nodelman, Christian R Shelton, and Daphne Koller. Learning continuous time bayesian networks. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 451–458. Morgan Kaufmann Publishers Inc., 2002b.
- Uri Nodelman, Daphne Koller, and Christian R Shelton. Expectation propagation for continuous time bayesian networks. In *Proceedings of the Twenty-first Conference on Uncertainty in AI (UAI)*, pages 431–440, Edinburgh, Scotland, UK, July 2005.
- James R Norris. *Markov chains*. Number 2. Cambridge university press, 1998.
- Manfred Opper and Guido Sanguinetti. Variational inference for markov jump processes. In *Advances in Neural Information Processing Systems*, pages 1105–1112, 2008.
- Michael K Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.
- Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines. Coda: Convergence diagnosis and output analysis for mcmc. *R News*, 6(1):7–11, 2006.
- Vinayak Rao and Yee W Teh. Mcmc for continuous-time discrete-state systems. In *Advances in Neural Information Processing Systems*, pages 701–709, 2012.
- Vinayak Rao and Yee W Teh. Fast MCMC sampling for Markov jump processes and extensions. *Journal of Machine Learning Research*, 14:3207–3232, 2013.
- Gareth O Roberts and Jeffrey S Rosenthal. General state space markov chains and mcmc algorithms. *Probability Surveys*, 1:20–71, 2004.
- Tore Schweder. Composable markov processes. *Journal of applied probability*, 7(2):400–410, 1970.
- Darren J Wilkinson. Stochastic modelling for quantitative description of heterogeneous biological systems. *Nature Reviews Genetics*, 10(2):122–133, 2009.